FAST AND EFFICIENT CALCULATION OF THE MULTILAYERED SHIELDED GREEN'S FUNCTIONS EMPLOYING NEURAL NETWORKS

Juan Pascual García, David Cañete Rebenaque, Fernando D. Quesada Pereira, Jose L. Gómez Tornero, and Alejandro Alvarez Melcón
Technical University of Cartagena
Campus Muralla del Mar s/n
30202 Cartagena, Murcia, Spain

ABSTRACT: *In this paper, neural networks are used to efficiently calculate the multilayered media boxed Green's functions needed in integral equation (IE) formulations. The analysis of complex multilayered shielded circuits is computationally time consuming, due to the need to calculate the multilayered-media boxed Green's functions. Using neural networks as radial basis function networks, the boxed Green's functions can be calculated quickly, thus greatly reducing the computational time associated with the analysis of practical circuits. Once the neural network is trained with a known set of pairs of inputs and outputs, new outputs are quickly calculated, thus increasing the efficiency of the IE method.*

Key words: *Green's functions; neural networks; multilayered media; boxed circuits*

1. INTRODUCTION

Shielded microwave circuits, such as monolithic microwave integrated circuits (MMICs), have attracted attention during recent years due to their features and advantages, including low cost, small size, and reliability. Due to the increase in the frequency operation and the high level of integration achieved, quasi-static or full-wave electromagnetic analysis is necessary. There are many full-wave methods for the analysis of shielded circuits and antennas. The finite-element method is a technique that has been successfully used, but is highly time-consuming. One of the most frequently used techniques is the integral equation method (IE) solved using the method of moments (MoM). In this method, the boxed Green's functions of the multilayered media are needed to solve the IE. The main problem of the IE is the slow convergence behavior of the modal series used to represent the relevant boxed Green's functions. Although the MoM is less time-consuming than the finite-element technique, it is still slow for developing computer aided design (CAD) tools that can operate with near real-time performance. The solution proposed in this paper to cope with the aforementioned problem is the use of neural networks to speed up the computation of the relevant boxed Green's functions.

Neural networks (NNs) have been used in recent years in many disciplines from signal processing to chemistry. As approximation function tools and nonlinear data interpolators, NNs can be used to solve a wide range of problems.

In this paper, the electric scalar-potential Green's function exact solution of the multilayered boxed media is evaluated by the technique exploited in [1] and then compared with the approximation performed by the NN.

NNs are composed by one or more hidden layers of nodes, called neurons, that perform a nonlinear operation. Each neuron is connected to several other neurons. Each connection has a weight that fixes the neuron importance in the net. The design of an NN consists of two stages. The first is the training step, where the NN learns from the data. In the second step, the new data are presented to the network to check the NN's generalization ability. Due to the simple structure of the NN, new outputs are quickly calculated. This ability means that these networks are able to be exploited in problems where others methods are highly time-consuming. Therefore, NNs are able to solve many different problems in electromagnetism [2].

In [3–5], multilayer perceptrons (MLPs) were used to evaluate the multilayered media Green's function integrals in order to transform the spectral domain. The authors called this method the neurospectral technique. This technique is limited to the analysis of nonshielded structures such as, for instance, a rectangular-patch antenna printed on a dielectric substrate with infinite transverse dimensions [5]. In [6], a kind of NN called the radial basis function network (RBFN) was used to calculate the coefficients of the discrete complex image method (DCIM) in layered media. In [6], the structure analyzed was also a printed circuit on a dielectric of infinite transverse dimensions.

In this paper, a study on the use of the RBFN for the calculation of the boxed Green's functions in multilayered media is carried out. To show the feasibility of this novel technique, the RBFN is applied in this paper to the calculation of the electric scalar-potential Green's function. The data used to train the NN are calculated using the technique presented in [1] for the accurate numerical evaluation of the multilayered-media boxed Green's functions. Once the NN is trained, new values of the real and imaginary parts of the Green's functions can be calculated more quickly than those using the exact method, with a desired level of accuracy. In section 2, a brief description of the method to compute the Green's functions in multilayered media is reviewed. Then, the NN used is shown, and different training methods are explained. Finally, the numerical results are presented, and a discussion on the best training method for this application is detailed.

2. GREEN'S FUNCTIONS IN MULTILAYERED MEDIA

The IE has been successfully used in the analysis of multilayered printed circuits. The main problem of the IE is the slow convergence behavior of the series used to represent the relevant boxed Green's functions. If a spectral-domain formulation is used, then they are represented by the following modal series:

$$G_B = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \tilde{G}_f(k_{x_m}, k_{y_n}) f(k_{x_m}, x, x') h(k_{y_n}, y, y'), \tag{1}$$

where $\tilde{G}$ is the spectral-domain Green's function, and f and h are combinations of sinusoidal functions. One procedure to accelerate the convergence of the series is to extract the quasi-static part of the spectral-domain Green's function from the dynamic part. The dynamic part converges quickly in comparison with the quasi-static part, and it can be summed directly in the spectral domain. In order to accelerate the convergence of the quasi-static part, it can be converted to the spatial domain, thus formulating an infinite series of spatial images.

In spite of the improvement in the convergence using this quasi-static extraction mechanism, there are situations where a large number of terms are needed to reach the desired level of accuracy. Therefore, suitable algorithms for series acceleration are needed. The algorithm for acceleration used in [1] consists of the application of the integration by parts technique to discrete sequences, hence, it was given the name, summation by parts technique. This algorithm is directly applied to the computation of the shielded Green's functions expressed with the modal series form of Eq. (1). After some operations, the original series is expressed in this new form [1]:

$$R_{M,N} = \sum_{m=M}^{\infty} \sum_{n=N}^{\infty} \tilde{G}_{m,n} f_m h_n = \sum_{i=1}^{\infty} \sum_{k=1}^{\infty} \tilde{G}_{M,N}^{(-i,-k)} f_{M-1}^{(i+1)} h_{N-1}^{(k+1)}, \tag{2}$$

where $(\tilde{G}_{m,n}^{(-i,-k)})$ are difference functions, defined as

$$\tilde{G}_{m,n}^{(-i,-k)} = \tilde{G}_{m+1,n}^{(\mathrm{i}+1,-k)} - \tilde{G}_{m,n}^{(-i+1,-k)}, \tag{3}$$

and $(f_n^{(i)})$ are sum functions, defined as

$$f_n^{(i)} = \sum_{k=n+1}^{\infty} f_k^{(i-1)}, \tag{4}$$

with $i = 1, 2, 3, \ldots$, and $k = 1, 2, 3, \ldots$. The sums given in Eq. (4) are obtained analytically by applying a simple geometrical summation formula. For $(i = 1)$, we first initialize the iterative algorithm as

$$f_n^{(1)} = h_n^{(1)} = \cos(nx). \tag{5}$$

Using the geometric summation formula, we easily obtain a closed-form expression for all other values of the index (i):

$$f_n^{(i)} = h_n^{(i)} = \frac{\cos\left\{(i-1)\frac{\pi}{2} + \left[n + (i-1)\frac{1}{2}\right]x\right\}}{2^{(i-1)}\left[\sin\left(\frac{x}{2}\right)\right]^{(i-1)}}. \tag{6}$$

The final Green's functions are calculated by making use of Eqs. (5) and (6) appropriately. As the observer point moves away from the source point, the number of iterations needed to achieve the given accuracy decreases. When the observer point is very close to the source point, however, more iterations are needed. In this case, the transformation to the spatial domain through a spatial-images series has proved useful. Also, the results show that the number of iterations in the algorithm remains constant with frequency [1]. This method can be effectively used for the accurate evaluation of the boxed Green's functions needed to train the NN, as described in the next section.

3. RADIAL BASIS FUNCTION NETWORK MODEL

Radial basis function network (RBFN) is one type of layered feedforward NN. These networks are composed of three layers, namely, the input layer, the hidden layer, and the output layer. The RBFN has been chosen in this paper over other kind of neural networks such as, for instance, MLPs, due to their simple structure, and fast training in comparison with MLPs. Furthermore, the RBFN, just like the MLP, has the ability to approximate any continuous function within an arbitrary accuracy. This is the case as long as a good choice of network parameters is taken. Overall, this behavior can be viewed as a multidimensional nonlinear curve-fitting [7].

The hidden layer of the network performs a nonlinear mapping, while the output layer performs a linear one. The mapping F from an input space of dimensionality n to an output space of dimensionality m, $F:\mathbf{R}^n \rightarrow \mathbf{R}^m$, is made by the following transformation:

$$F(\mathbf{x}) = b + \sum_{k=1}^{J} w_k \cdot \varphi(\|\mathbf{x} - \mathbf{c}_k\|), \tag{7}$$

where $\mathbf{c}_k$ is the center of the k^{th} radial basis function, $\mathbf{x}$ is the input data, b is a constant or bias, w_k is the weight of the k^{th} neuron, and J is the number of neurons. Moreover, $\|\cdot\|$ is the Euclidean norm, and $\varphi(\cdot)$ is the nonlinear function used to implement the nonlinear mapping of the hidden layer.

In our problem, the input data is a set of spatial coordinates that represents the transverse spatial positions of the observer and source points inside a metallic box. The remaining spatial longitudinal positions (z, z') are discrete, as imposed by the multilayered nature of the circuit. They are directly included in the multilayered-media spectral-domain Green's functions [8]. With these considerations, the input space is 4D. All source and observer points are placed in a uniform net that discretize the cross section of the metallic box. The output data is 2D, consisting of the real and imaginary parts of the electric scalar-potential Green's function calculated for each source point at each observer point.

For an optimum performance of the RBFN, the most critical choice is the proper selection of the radial-basis centers. There are many nonlinear functions that can be used as radial-basis functions. In our case, the next Gaussian multivariate function has been used:

$$\phi(\|\mathbf{x} - \mathbf{c}_k\|) = \exp - \frac{\|\mathbf{x} - \mathbf{c}_k\|^2}{2\sigma^2}, \tag{8}$$

where $\mathbf{c}_k$ is the center of the Gaussian function and σ^2 is the Gaussian variance. The centers and variances are the parameters to be computed by the chosen training algorithm. A known set of input and output data is delivered to the RFBN at the input in order

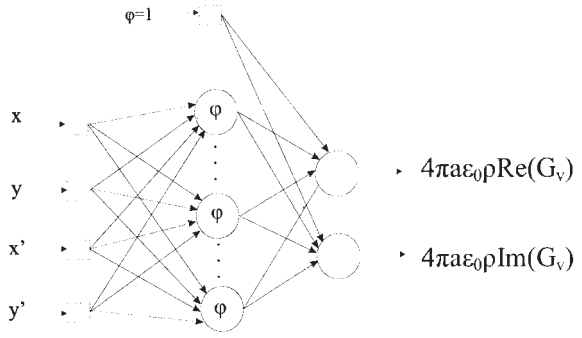

Figure 1 Neural network structure

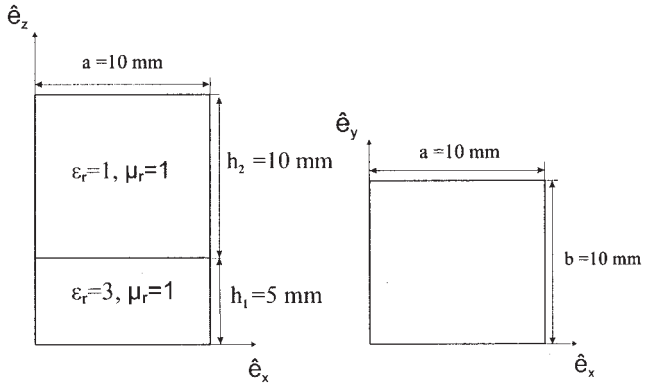

Figure 2 Shielded microstrip multilayered structure

to train it. This process leads to the selection of the appropriate centers, variances, and weights. After the training phase, the neuron centers, variances, and weights will remain fixed. At the output, each neuron is multiplied by the respective weight, and all are summed up with the corresponding bias to obtain the final output values (see Fig. 1).

3.1. Training Algorithms

Three different RBFN training algorithms have been tested, as applied to the calculation of the boxed Green's functions. The first strategy is the fixed centers selected at random [7], which consists of choosing the centers randomly from the training data. The second one is a self-organized selection of centers called k-means clustering [7]. The aim of this procedure is to put the centers in regions of the whole input space where the most important data are present. For these first two algorithms, the standard deviation of each Gaussian radial basis function is the mean of the nearest p-centers [9]:

$$\sigma_k = \frac{\alpha}{p} \sum_{i=1}^{p} d_{\tau(i)}, \tag{9}$$

where α is a parameter to be set heuristically, and $d_{\tau(1)} < d_{\tau(2)} < \ldots < d_{\tau(p)} < \ldots < d_{\tau(M-1)}$. Moreover, the distance between centers is calculated using the Euclidean norm. For both the first and second learning strategies, we have carried out the calculation of the weights for the linear layer using a well-known linear algorithm, namely, the least mean square technique (LMS). LMS is a supervised learning strategy because the values of the outputs influence the calculation of the weights at the linear output layer. Other techniques, such as fixed centers selected at random and k-Means Clustering, are unsupervised procedures.

The last learning strategy investigated in this paper is the classical orthogonal least squares (OLS) training algorithm [10]. In this procedure, the centers are selected from an initial set of centers, generally a subset of the input data. In our case, we have taken the subset formed with all the training input data set as the initial centers set. In OLS training, we first calculate the regression matrix. The columns of the regression matrix (**P**) are the regressors vectors, resulting from the evaluation of the radial basis nonlinear function at every training input datum for a certain specific center. The regressors matrix **P** is orthogonalized applying the Gram–Schmidt method. Then, **P** is transformed into a product of two matrices, namely, **W** (composed of orthogonal columns) and a triangular matrix that we call **A.** The RBFN operation can be expressed as $\mathbf{d} = \mathbf{Pw} = \mathbf{Wg}$, where **d** is the desired output vector. Each step of the orthogonalization method implies the computation of a column of matrix **W**, which implicitly becomes the selection of a center. Finally, the neuron weights **w** are calculated by making use of the triangular matrix **A**, in the following way: $\mathbf{Aw} = \mathbf{g}$. When OLS is used, the variance has to be preset at the beginning of the algorithm.

In [11], it is demonstrated that the centers set given by the OLS classic algorithm is not the most compact solution when a non-orthogonal basis is used. In spite of this, OLS has proved to be useful in many applications, and as the results will show, it leads to a better performance than the fixed centers selected at random and the k-means clustering algorithms.

3.2. Training and Testing Data Set

The square metallic box studied in this paper, as shown in Figure 2, is divided into a uniform net of equidistributed spatial points (see Fig. 3). The input data is a set of transverse spatial coordinates, where each input vector is composed of four coordinates containing the source (x', y') and observer transverse positions (x, y). Each coordinate is normalized with respect to the size of the box. Due to this normalization, the coordinates can vary from 0 to 1.

The output of the neural network for an input vector is the real and imaginary parts of the electric scalar potential: $Re(G_v)$ and

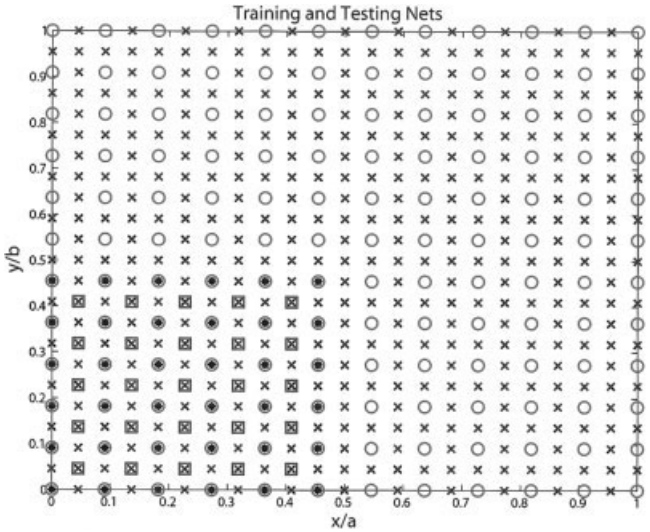

Figure 3 Training and testing Data: testing-source data (square points), testing-observer data (cross points), training-source data (asterisk points), and training-observer data (circle points). Box dimensions are 10×10 mm

$Im(G_v)$. The frequency and the box dimensions are set as fixed parameters. Due to the problem symmetries, we can divide the net into four quadrants and compute the potential for the sources belonging to only one quadrant, thus saving computational time. For every source point, the potential is evaluated in all the observer points of the net, including the other three quadrants. In order to avoid the abrupt behavior of the Green's function closed to the source, the singularity of the electric scalar potential is properly extracted [1]. With the above mechanism, if the source is placed in a different quadrant from the one where the RBFN was trained, the potential is extracted from the symmetric source point placed at the first quadrant.

The net depicted in Figure 3 was generated to train the RBFN. For every source point, the electric scalar-potential Green's function was calculated for every observer point, including the one placed at the source. In Figure 3, the testing input points used to check the generalization properties of the RBFN are also shown. Both the observer and source points of the testing net are placed in intermediate positions in relation to the spatial points of the training net.

4. RESULTS

For the calculation of the electric scalar-potential Green's function G_v, we have taken a metallic box of dimensions 10 × 10 mm (Fig. 1). The potential was calculated for a frequency of 15 GHz (λ/2 box size), and 30 GHz (λ box size). The RBFNs were trained with the three algorithms explained in the previous section, and for the two frequencies selected.

For training each RBFN, a 12 × 12 spatial-points net was generated. Due to the symmetry of G_v, the sources were restricted to a 6 × 6 points subset (first quadrant of the box). The electrical scalar-potential Green's function was computed at every point of the 12 × 12 net. Then, 5184 patterns were generated for training the RBFN (see Fig. 3). The testing set consisted of 9625 patterns, as shown in Figure 3. The normalized mean square error (NMSE) was used to measure the RBFN error in the computation of the output as follows:

$$NMSE = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{2}[y_{ij} - \hat{y}_{ij}]^2}{\sum_{i=1}^{N}\sum_{j=1}^{2}[y_{ij} - \bar{y}_j]^2}}. \quad (10)$$

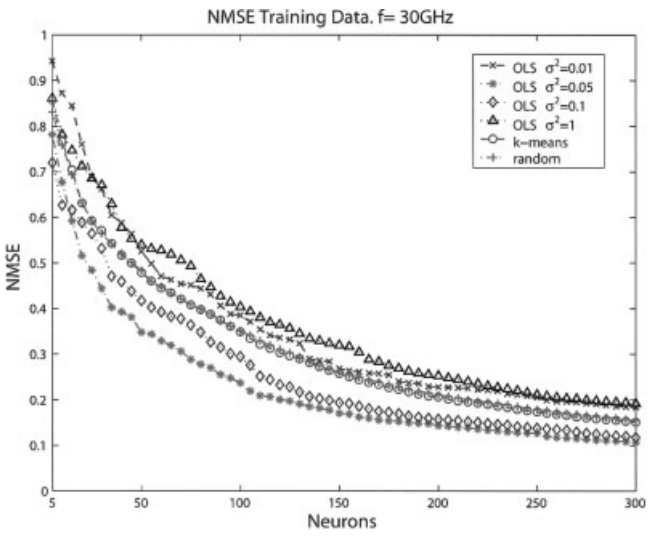

Figure 4 Normalized mean square error of the training data set for the OLS, k-means clustering, and random selection of centers procedures (frequency of the electric scalar potential Green's function G_v is 30 GHz)

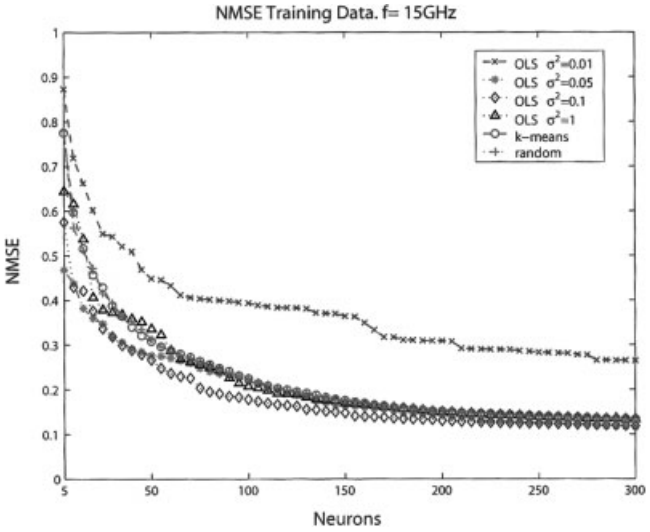

Figure 5 Normalized mean square error of the training data set for the OLS, k-means clustering, and random selection of centers procedures (frequency of the electric scalar potential Green's function G_v is 15 GHz)

In Figures 4 and 5, the NMSE values of the training data for the RBFNs designed are displayed. In each figure, the performance of six different RBFNs are depicted: four RBFNs are trained with OLS and different variances, one is trained with k-means clustering, and finally one is trained with the fixed centers selected at random algorithm. The NMSE is depicted for every five neurons, from five neurons up to 300 neurons. The NMSE in the case of the random centers selection is the average of ten different runs. At each run, different random centers are generated.

For all the selection-centers procedures, it is apparent that as the number of neurons in the hidden layer grows, there is a corresponding reduction of the NMSE. In a practical application, a maximum error allowed in the network can be preset before the training begins. The network adds neurons until the threshold is reached. For the problem treated in this paper, a proper threshold value has been found to be 0.15.

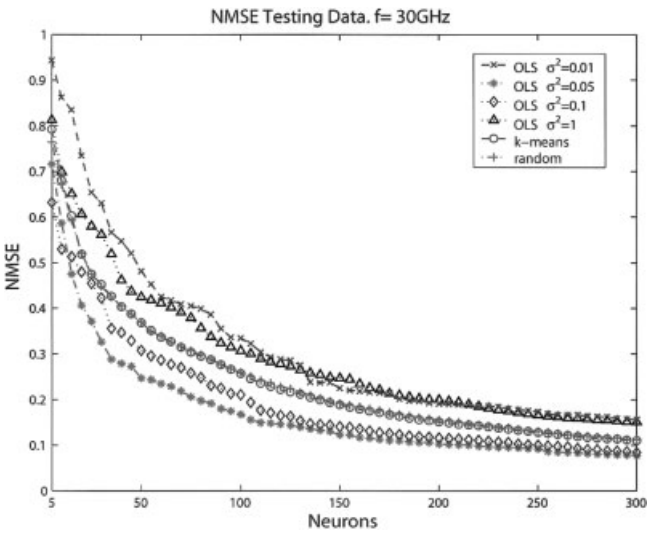

Figure 6 Normalized mean square error of the testing data set for the OLS, k-means clustering, and random selection of centers procedures (frequency of the electric scalar potential Green's function G_v is 30 GHz)

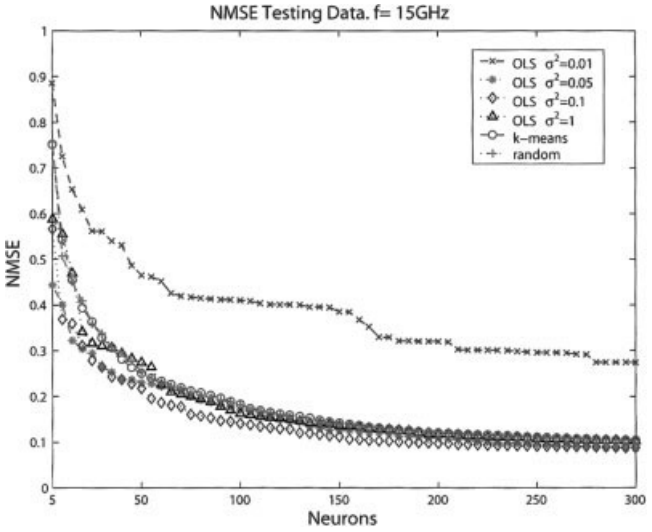

Figure 7 Normalized mean square error of the testing data set for the OLS, k-means clustering, and random selection of centers procedures (frequency of the electric scalar potential Green's function G_v is 15 GHz)

In the RBFNs designed to approximate G_v at 30 GHz, the best results correspond to the OLS algorithm with variances between 0.05 and 0.1 (see Fig. 4). It can be seen that the maximum error allowed is reached with 200 neurons. The next-best results correspond to the k-means clustering and fixed centers selected at random. In these cases, 300 neurons are needed to surpass the preset error threshold of 0.15. The worst performance is achieved by OLS with the extreme values of variance 0.01 and 1.

In Figure 5, the error corresponding to the RBFNs trained to approximate G_v at 15 GHz is shown. Note that high error levels are reached by OLS with a variance of 0.01. On the other hand, OLS with variances between 0.05 and 0.1 has a slightly better performance than the k-means clustering and fixed centers selected at random. Moreover, OLS with a variance of 1 reaches results similar to those of these two last procedures. Error values below

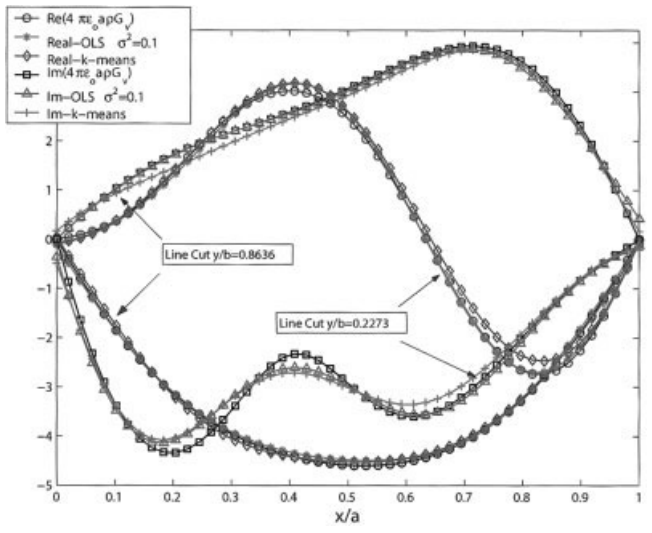

Figure 8 Exact potential and two 300-neuron RBFN approximation: one RBFN was trained using OLS with $\sigma^2 = 0.1$ and the other one was trained with the k-means algorithm; the source point is placed at $(4.091/a, 3.182/b)$; the observer points are placed in a line cut at $y/b = 0.2273$ and in a line cut at $y/b = 0.8636$; frequency $f = 30$ GHz

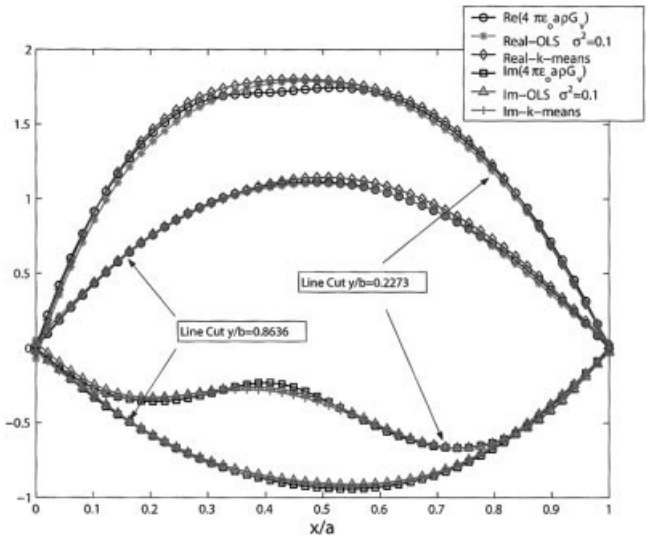

Figure 9 Exact potential and two 200-neuron RBFNs approximation: one RBFN was trained using OLS with $\sigma^2 = 0.1$ and the other one was trained with the k-means algorithm; the source point is placed at $(4.091/a, 3.182/b)$; the observer points are placed in a line cut at $y/b = 0.2273$ and in a line cut at $y/b = 0.8636$, frequency $f = 15$ GHz

0.15 are achieved with 150 neurons if OLS with variances between 0.05 and 0.1 is used to train the network. If fixed centers selected at random or k-means clustering is used, the maximum error allowed is reached with 200 neurons. The variances of the neurons in OLS are the same for all of them, and they are chosen heuristically. This is the most important weakness of this procedure. But this weakness is shared by the k-means clustering or fixed centers selected at random strategies. In these last two techniques, another parameter, namely, the number of nearest centers which are needed to compute the Gaussian function variance, has to be preset.

As seen from Figures 4 and 5, the OLS performance depends on the initial preset variance. The extreme values of the variance need to be avoided because they produce too flat or too peaked Gaussian functions. However, the OLS procedure, with the appropriate variance values, has achieved lower NMSE levels than the ones obtained when a random set of centers is selected, or when k-means clustering is used. Fixed centers selected at random usually leads to a worse performance and due to the near-linear

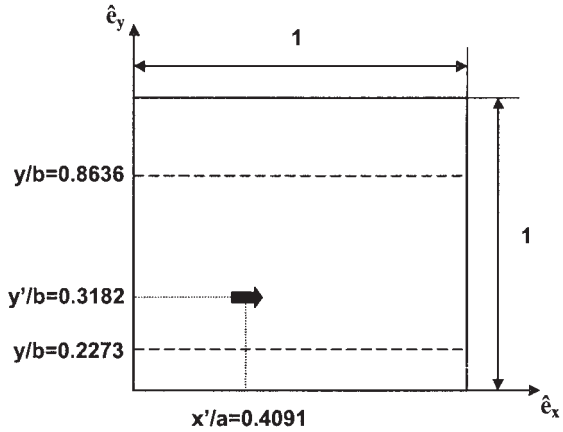

Figure 10 Source placed at $(4.091/a, 3.182/b)$, a line cut placed at $y/b = 0.8636$, and a line cut placed at $y/b = 0.2273$

TABLE 1 Computational Time to Evaluate the Electric Scalar Potential Green's Function in 5184 Points with 300, 200, and 150 Neurons RBFN (Each Value is a 20 Runs Average)

	300 Neurons	200 Neurons	150 Neurons
Time (seconds)	0.96	0.59	0.42
Times faster than exact solution	48.7	79.3	110.9

dependency introduced by centers placed too close, numerical ill-conditioning in the weights estimation problem is generated. The resulting RBFN is always too large for a given error, or it works poorly if a fixed number of neurons is adopted. Because of the structure of the input data in the Green's function approximation problem (there are no clusters in the input space), the self-organized selection of centers in the k-means clustering procedure does not represent a clear advantage. In spite of this, k-means clustering allows a better selection of centers than the fixed centers selected at random procedure.

The testing data NMSE of the RBFNs trained are depicted in Figures 6 and 7. The results show that the RBFN generalizes well for new inputs. The designed RBFNs are able to interpolate output values in a correct way. Again, the OLS algorithm is the best approximation to the electric scalar potential Green's function G_v.

In Figures 8 and 9, it is shown qualitatively how the RBFNs approximate the smooth variations of the exact electric scalar potential Green's function for a source point placed in the testing net. In Figure 10, the source point and two observer line cuts, where the Green's function is calculated, are displayed. The line cut placed at $y/b = 0.2273$ has been chosen to show the RBFN performance near the source. On the other hand, the line cut placed at $y/b = 0.8636$ is placed far from the source. Each line cut consists of 50 equidistributed observer points that belong to the corresponding line of the testing net. Therefore, both the source and observer points are new input data to the neural network. Although the approximation made by the k-means clustering algorithm is closed to the exact solution, OLS leads to better results, as shown in Figures 4 and 7.

Once the RBFN is trained, new outputs can be fast calculated using the neural network. In Table 1, the time required to compute the training data with 150, 200, and 300 neurons is shown. The computational time consumed to calculate the exact solution was 46.8 s. The OLS training time for a 300-neuron RBFN was 817.4 s. The training time took 41.612 s for k-means clustering. Finally, it took 14.67 s for the random selection of centers algorithm. The computer used to carry out these simulations was a Pentium IV with 3.06-GHz processor.

5. CONCLUSION

This paper has presented the use of neural networks (NNs) for the calculation of multilayered-media boxed Green's functions. Once the radial basis function network (RBFN) is trained, new potential values of any source point can be quickly calculated. The neural network takes as an input the spatial coordinates of the source and observer points, with a fixed frequency. The performances of three different RBFN learning strategies are studied. The RBFN output values for the training and testing data show that the network is a good model for the calculation of the electric scalar-potential Green's function. The 200-neuron RBFN was nearly 80 times faster than the exact method used to carry out the computations. The 300-neuron RBFN was more than 50 times faster than the exact procedure. The orthogonal least squares (OLS) algorithm leads to the best results. The variance that allows the best performance for this learning strategy comprises values from 0.05 to 0.1. This leads to networks with only 200 neurons necessary to achieve the selected error level. The results presented show that the RBFN is a useful tool in the design CAD tools for microwave circuits, where hundreds of Green's functions have to be calculated. The time saved when hundreds of Green's functions are calculated using the RBFN makes the training time negligible, even when OLS is used.

REFERENCES

1. A.A. Melcon and J.R. Mosig, Two techniques for the efficient numerical calculation of the green's functions for planar shielded circuits and antennas, IEEE Trans Microwave Theory Tech 48 (2000), 1492–1504.
2. C. Christodoulou and M. Georgipoulos, Applications of neural networks in electromagnetics, Artech House, 2001.
3. R.K. Mishra and A. Patnaik, Neurospectral computation for input impedance of rectangular microstrip antenna, Electron Lett 35 (1999), 1691–1693.
4. R.K. Mishra and A. Patnaik, Neurospectral analysis of coaxial fed rectangular patch antenna, IEEE Antennas Propagat Soc Int Symp 2 (2000), 1062–1065.
5. R.K. Mishra and A. Patnaik, Designing rectangular patch antenna using the neurospectral method, IEEE Trans Antennas Propagat 51 (2003), 1914–1921.
6. E.A. Soliman, M.A. El-Gamal, and A.K. Abdelmageed, Neural network model for the efficient calculation of green's functions in layered media, Int J RF Microwave CAE 13 (2003), 128–135.
7. S. Haykin, Neural networks: A comprehensive foundation, Prentice Hall International, New Jersey, 1999.
8. J.R. Mosig, Integral equation technique, Wiley–Interscience, New York, 1989.
9. F. Schwenker, H.A. Kestler, and G. Palm, Three learning phases for radial-basis-function networks, Neural networks 14 (2001), 439–458.
10. S. Chen, C.F.N. Cowan, and P.M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Trans Neural Networks 2 (1991), 302–309.
11. A. Sherstinsky and R.W. Picard, On the efficiency of the orthogonal least squares training method for radial basis function networks, IEEE Trans Neural Networks 7 (1996), 195–200.